Brique B.D.L : **Gestion de Projet Logiciel**

Editeur graphique d'aide à la description d'une architecture logicielle

Revue de Planification

Yann Barthélémy
Audran Le Baron
Olivier Jennes
Pierrick Lerey
Alexandre Tisserant

Moniteur : **Linda Djezzar**

Expert : **Sylvie Vignes**

Année scolaire 2002-2003 / Trimestre 1

Table des matières

1 Les fondements de notre logiciel

La définition d'une architecture logicielle est une étape déterminante dans la conception d'un logiciel. Après la phase d'analyse des besoins, en décrivant a priori le logiciel en termes de modules logiciels et d'interactions entre ces modules, le concepteur en établit en quelque sorte un "plan de construction", en se dégageant des détails techniques d'environnement. L'architecture logicielle est donc une spécification abstraite décrivant les spécifications fonctionnelles ou non fonctionnelles des modules et les règles d'interactions nécessaires à la réalisation globale de l'application.

Ce développement par composition favorise la réutilisation de composants. Un autre intérêt est de dégager des architectures génériques de familles de produits, c'est à dire plusieurs versions d'une seule application.

Le besoin de description d'architecture a fait apparaître de nombreux langages ADLs (Architecture Description Languages). Des divers formalismes proposés, il ressort trois concepts essentiels : le composant, le connecteur et la configuration.

Le composant est une entité logicielle autonome définie par son interface et son descripteur interne. L'interface décrit les services fournis ou requis par le composant ; la description interne décrit le fonctionnement de l'implémentation. Le connecteur est un élément d'architecture décrivant les règles d'interactions entre les composants.

La configuration correspond à un assemblage particulier, une instanciation des composants et connecteurs pour former l'application. Elle décrit donc les règles d'assemblage des composants.

Récemment, devant le foisonnement des ADLs, il a été proposé à la communauté scientifique de se baser sur un langage générique commun **xADL** pour décrire une architecture logicielle. **xADL** est un langage XML, dont la grammaire est adaptée aux descriptions évoquées ci-dessus.

L'objectif de ce projet est de développer un outil d'aide à la conception d'architecture qui permette une saisie graphique de boites et liens représentant les composants et les connecteurs. L'outil devra permettre la génération du fichier **xADL** correspondant en s'assurant de sa conformité à la grammaire du langage. Ce fichier **xADL** doit pouvoir être édité pour que des compléments de descriptions y soient incorporés. Pour le concepteur de l'application il est aussi intéressant d'avoir la description arborescente de ce fichier.

Nous allons donc nous attacher à développer ce logiciel que nous avons baptisé **SofAA** pour **Sof**tware **A**rchitecture **A**ssistant.

FIG. 1 – logo sofaa

2 Définition du cahier des charges

- SofAA est un outil graphique qui comprend 4 types d'objets :
  1. Les composants
  2. Les connecteurs
  3. Les liens
  4. Les groupes
- En sortie, SofAA génère un fichier XML qui respecte les schémas définis par le langage de description d'architecture **xADL**.
- Les sources de **SofAA** sont évolutives dans un souci de mise à jour et d'upgrade futurs.
- **SofAA** est en quelques sortes un guide pour son utilisateur dans la mesure où notre logiciel lui propose tous les champs à renseigner.
- L'ergonomie de **SofAA** doit rendre notre logiciel accueillant et facile d'utilisation.
- **SofAA** est "branché" en sortie avec une visionneuse XML afin de permettre à l'utilisateur d'avoir une vue sous forme d'arbre du fichier généré.

3 L'environnement et les contraintes d'utilisation

- Nous allons développer **SofAA** sur la plateforme JAVA de **SUN®** basée sur le "**jdk 1.3**" afin de pouvoir utiliser le paquetage "**Swing**" spécifique à la création d'interfaces graphiques.
- Nous allons utiliser le parseur XML (SAX) de la classe java "**Xerces**".
- Notre produit bénéficiera donc d'une grande portabilité du fait de sa réalisation sur une plateforme JAVA.
- **SofAA** ne devrait pas souffrir de problèmes liés à ses performances car la génération de fichiers XML est d'une nature relativement simpliste.
- **SofAA** ne devrait pas avoir des besoins de mémoire spécifiques car la machine virtuelle JAVA va s'en charger.

4 Catégories d'utilisateurs

- Dans une optique de court terme, le but de **SofAA** est de faciliter la génération de fichiers **xADL** aux **chercheurs qui connaissent et utilisent déjà de langage**.
- Dans le cadre des méthodes de développement génie logiciel mises en œuvres par les responsables de projets ou par les développeurs, **SofAA** constitue un excellent outil d'aide à la répartition des tâches lors de la phase de planification.
- A plus long terme, l'optique est plus générique. On peut imaginer que **SofAA** puisse être intégré dans une suite applicative plus complète de développement génie logiciel.

5 La livraison finale

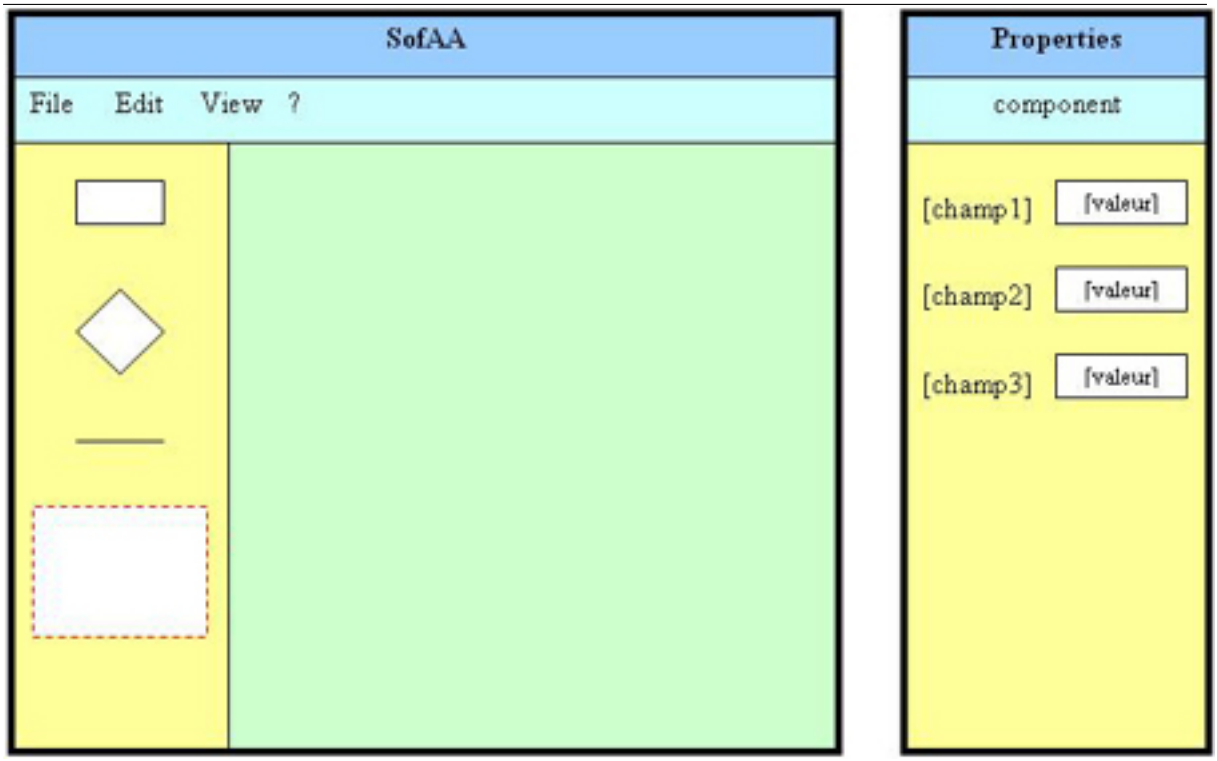
- **SofAA** comprend un ensemble de paquetages JAVA qui s'installent d'une manière totalement automatisée.
- Un script SHELL permet de vérifier les variables d'environnement et de les modifier pour chaque utilisateur. Il lance ensuite notre logiciel **SofAA**.
- Un manuel d'utilisation au format HTML sera également disponible.
- Nous allons créer une page de "man" pour les environnements UNIX.
- Nous mettrons à la disposition de l'utilisateur des fichiers d'exemples **xADL** générés par **SofAA**.

6 Définition du Manuel d'utilisation

6.1 L'interface graphique de SofAA

- **SofAA** comprend 2 fenêtres.

- Dans la fenêtre principale, l'utilisateur va trouver un menu comportant les rubriques "File", "Edit", "View" et " ?" qui reprennent les sous-rubriques habituelles.



– Les 4 types d'objets que nous avons définis (les composants, les connecteurs, les liens et les groupes) pourront être cliqués-déplacés depuis la partie gauche de la fenêtre vers le plan de travail situé sur la droite de cette même fenêtre.
– Dans la fenêtre auxiliaire que l'on a qualifié de "**fenêtre de propriétés**", les champs spécifiques au type de l'objet sélectionné par l'utilisateur vont s'afficher afin qu'il puisse les consulter ou les renseigner.

6.2 Le fichier de "man" pour les environnements UNIX

```
SofAA            USER COMMANDS            SofAA

NAME
SofAA - software architecture assistant


SYNOPSIS
SofAA [file.xml]


DESCRIPTION
SofAA is a graphic tool for designing xADL
compliant software architecture. It manipu-
lates XML files in accordance with xADL sche-
mas

file.xml - the XML file to edit
```

7 La répartition des rôles

Alexandre TISSERANT sera le responsable de l'équipe de développement du projet **SofAA**. **Yann BARTHÉLÉMY** contribura à la réalisation des test. **Olivier JENNES**, également. **Audran LE BARON** sera le responsable CVS. **Pierrick LEREY**, le responsable de la documentation.

8 Les ressources utilisées

- ***emacs*** sera notre éditeur de sources.
- ***LaTeX 2ε*** sera notre éditeur de documentation.
- ***Microsoft® PowerPoint®*** sera notre éditeur de présentations.
- ***CVS*** sera notre gestionnaire de versions.
- Le ***jdk 1.3 de SUN®*** sera notre plateforme de développement
- "***Sax parser***" du paquetage ***Xerces*** sera notre parseur XML.
- ***Javadoc*** sera le générateur de documentation de nos sources java.
- ***Bugzilla*** sera notre "Bug Tracking System".

9 Planification

phase	estimation	réalisé
Planification (revue le 11/10/2002)	12 personnes.heures	16 personnes.heures
Analyse (revue le 04/11/02)	12 personnes.heures	
Conception (revue le 22/11/2002)	24 personnes.heures	
Implémentation et tests (revue le 16/12/2002)	100 personnes.heures	
Total	148 personnes.heures	16 personnes.heures